Larger Internal Ptrs Plan

Changes required in system

Fri, Oct 18, 2002

In order to add the feature of permitting larger internal ptr formats than 4 bytes, certain changes must be made to the system code. This note describes the required changes.

PReqDat.c

This routine is called to support a Classic "network data request," which ignores all idents in the array of idents within the data request that do not refer to the local node.

PSLN

Evaluates NIDLOC and ANSNB by looping over the listypes.

INZBLOCK

In the request block, NPTRS is set to local variable NPT, but this field is likely to be only a diagnostic that is not used by any other code.

Request block field ANSWERS computation assumes use of NPT << 2 to advance past the space needed for the internal ptrs array.

After looping over the listypes in the request, each time invoking PReqDGen to generate the internal ptrs, NPTRS is again set to the number of longwords by which the ptr to internal ptrs was advanced. This is done to update NPTRS to reflect the number of local idents as opposed to the default total number of idents, whether local or not. After setting NPTRS, it is checked against NPT, which it should match, unless there is an error detected by one of the ptr-type routines.

PReqData

After calling PSLN to work out the number of bytes needed to contain the reply data, or the #bytes of answers, NPT is set to NIDLOC * NLT, using the number of local idents noticed by PSLN. This value of NPT is used to derive the space needed in the request block for the internal ptrs array, as NPT << 2. Then INZBLOCK is called.

Changes required:

We can think of the NPT variable as a count of the number of longwords required for the internal ptrs array, rather than the number of internal ptrs. With this change in meaning, much of the existing code will work without modification.

Modify PSLN so that it also computes NPT carefully, checking the number of longwords required for an internal ptr for each listype in the request. This can be done via a call to a new routine LTTPSZ, with the single argument listypeID. This function returns the number of longwords required for an internal ptr.

Remove the code in PReqData that, after invoking PSLN, sets NPT to NIDLOC * NLT. This is ok, because the new PSLN will have computed NPT correctly.

ReqDat.c

ReqDataCommon2

LRBSZ (local request block size) uses NPT << 2 to get space needed for internal ptrs.

ReqDataCommon3

NPTRS set to NPT, but NPTRS is probably not used by any code.

EXTANS field assumes space needed for internal ptrs is NPT << 2.

ReqDataCommon

Loop over listypes, check for errors.

NPT set to totalPointersRequired, which comes from NID * NLT, just before calling ReqDataCommon3.

Changes required:

Assume that NPT can adopt the meaning of the number of longwords needed for the internal ptrs array, not the number of internal ptrs.

Add code to the loop over listypes in ReqDataCommon so that it calls LTTPSZ to get the size needed per internal ptr, and in this way calculates totalPointersRequired. Then, of course, remove the code that derives totalPointersRequired from NID * NPT.

ACReq.c

My guess is that NPTOTAL can be considered a count of the number of longwords needed for the internal ptrs array, and that will likely greatly reduce the number of changes required. In NSERVER, call LTTPSZ to get the value that must be multiplied by identCount when adding to NPTOTAL.

Consider this means of communicating the size of an ident from the "compilation" phase to the "execution" phase. Use the high byte of the PPI short to hold the number of longwords occupied by each internal ptr used by the DRB entry. rather than describing it this way, define two byte-size fields, the first called PSZ and the second called PPI. PPI used to be a short, but it can as well be a byte. The value in PSZ is the value returned from a call to LTTPSZ(listypeID) used by that entry.

This new PSZ byte will be used by ACUpdate to advance the ptr to the array of internal ptrs processed for each DRB. It will advance this ptr by PSZ * NID * 4.

The reason that we cannot change the meaning of NID to be the number of longwords in the internal ptrs array is that NID * NBY is used to advance the ptr to the answers being updated. NID has to remain the number of idents, which is nearly always 1.

ACReq uses a trick already to add additional space for some special internal ptrs that have to do with averaging readings and also for 7.5 Hz time-stamped requests. But I hope that we do not have to fold in that special logic with this new stuff. Those cases should be supported by listypes for which the ptr-type# is less than 32 and therefore this support for expanded-size internal ptrs is not used.

I think when all the details are worked out that this should not be difficult.